You have been CAUTE! Early Detection of Compromised Accounts on Social Media

Courtland VanDam*
MIT Lincoln Laboratory
Lexington, MA, USA
courtland.vandam@ll.mit.edu

Farzan Masrour
Michigan State University
East Lansing, MI, USA
masrour@cse.msu.edu

Pang-Ning Tan
Michigan State University
East Lansing, MI, USA
ptan@cse.msu.edu

Tyler Wilson
Michigan State University
East Lansing, MI, USA
wils1270@msu.edu

Abstract—**Detection of compromised social media accounts is an important problem as the compromised accounts can be exploited by hackers to spread false and misleading information. In particular, early detection of compromised accounts is essential to mitigating the damages caused by the hackers' posts, which may range from victim shaming to causing widespread public panic and civil unrest. This paper proposes CAUTE, a deep learning framework that simultaneously learns the feature embeddings of the users and their posts in order to identify which, if any, of their posts were written by a different person, i.e. a hacker. Using Twitter as an example of the social media platform, CAUTE learns a tweet-to-user encoder to infer the user features from tweet features and a user-to-tweet encoder to predict the tweet content from a combination of the user features and the tweet meta features. The residual errors of both encoders are then fed into a fully-connected neural network layer to detect whether a post was published by the specified user or by a hacker. Experimental results showed that the features learned by CAUTE are more informative than those generated by conventional representation learning methods. Additionally, CAUTE outperformed several state-of-the-art baseline algorithms in terms of their overall performance and can effectively detect compromised posts early without generating too many false alarms.**

I. Introduction

Social media has gained significant popularity over the last decade with more than two-third of all Americans using some form of social media according to a recent study[1]. Unfortunately, due to the wealth of personal information available, social media platforms have also become an attractive target for hackers. According to a 2017 Pew Center study, 13% of Americans have experienced a compromised social media account [1]. The compromised accounts are often used to spread misleading information, including spam and false information. Victims of compromised accounts may lose their friends [2], or worse still, if a reputable account is compromised, such as a news agency's Twitter account, the hackers' posts can lead to public panic and volatile fluctuations of the stock market [3]. Thus, accurate and early detection of compromised accounts is essential to protect social media users from such malicious threats and mitigate their potential damages.

Compromised account detection is a challenging problem for several reasons. First, a compromised post, i.e. a post written by the hacker, may resemble a normal post for another user. For example, a hacker may lie about the user's sexuality to spread a rumor, but such posts are not unusual for some users who discuss their sexuality openly on social media. Detecting compromised posts is therefore more difficult compared to other tasks such as spam detection as spam posts tend to have features that are more easily distinguishable from the regular posts of normal users. Although there has been previous research to address this challenge by learning the behavioral profile of each user [4], [5], [6], they require a significantly large sample size to build a reliable user representation. If the number of tweets used to generate the profile is too low, it will flag too many false alarms. If the number of tweets needed is too high, then it cannot be effectively used for early detection. In addition, it is also computationally expensive to generate a profile for every user.

Second, the raw features of the social media data (e.g., the text messages and other meta-features of user tweets) are often sparse and noisy. For example, tweet messages tend to be short, i.e., 140 characters long, and contain typos and other non-standard lexical variations, making it difficult to effectively use them for training a robust compromised account detection model. While there has been previous research focusing on deriving reliable feature representation for detecting compromised accounts, they are mostly limited to linear (e.g., using principal component analysis [7]) and unsupervised [8] learning methods. As a result, the derived features may not be optimal for compromised account detection purposes.

To address these challenges, we propose CAUTE (Compromised Account User Tweet Encoder), a deep learning framework that simultaneously learns the nonlinear embeddings of users and their posts, and detects whether a post is compromised. CAUTE considers both lexical and meta features of a tweet to determine whether it was posted by the genuine account holder or a hacker. It accomplishes this by learning a pair of encoders to transform the raw features into more informative features while reducing their sparsity and noise. The first encoder, *tweet2user*, learns a latent representation to help transform the tweet features into user features while the second encoder, *user2tweet*, learns a representation that helps predict the content of a tweet from the user's features and tweet's meta-features. We hypothesize that, if a user is indeed the author of a post, then the errors associated with the tweet2user and user2tweet encoders for

[1]http://www.pewinternet.org/fact-sheet/social-media/

a given (user, tweet) input pair are expected to be low. Otherwise, the errors are likely to be high if the tweet was composed by another user (hacker). Instead of applying some arbitrary threshold, the residual errors of the encoders are fed into a fully-connected neural network layer, *res2class*, to predict whether the post is compromised for that given user. In principle, since the user features can be derived from the user's profile, CAUTE is applicable even in a cold start scenario, when the user has not posted any tweets, unlike other existing methods. However, unless the user profile includes meaningful information about the topics of interest to the user, it may not be sufficient to train a robust model. To enhance CAUTE's performance, the user features can also be derived from a small set of their initial tweets. Unlike other approaches that require a large training set to learn a reliable profile of the users, our empirical results suggest that CAUTE can effectively identify compromised posts after observing only as few as 10 of their initial tweets.

The main contributions of this paper are as follows:

- We propose CAUTE, a deep learning framework that can detect compromised posts by modeling the user and post features simultaneously.
- We show that the nonlinear embeddings derived by the tweet2user and user2tweet encoders are informative predictors of compromised posts.
- We demonstrate that CAUTE outperforms state-of-the-art baseline algorithms in terms of their accuracy as well as their ability for early detection without generating a large number of false alarms.

The remainder of this paper is as follows. In Section II, we summarize previous research on compromised account analysis and detection. The compromised post detection problem is formally defined in Section III. Section IV describes the CAUTE framework. Experimental results are presented in Section V, followed by our conclusions in Section VI.

II. Related Work

Previous works have considered three general approaches to compromised account detection: (1) creating a behavior profile for each user, (2) measuring distances between sets of activities, or (3) summarizing features at the user level. In the behavior profile approach, a subset of the users' activities are used to generate the user's behavior profile [4], [5]. Withheld activities are compared to this profile to determine whether the activity is anomalous. For example, the behavioral profile used by Nauta [5] for anomalous post detection was constructed from meta-data, such as post-time or application, and contextual features, such as hashtags or mentions in the posts. Egele et al. detected spam campaigns by finding multiple accounts with similar anomalous posts [4], [9]. The behavioral profile approach has also been applied to detect other anomalous behavior; e.g. login and browsing activities [10], [11].

The distance-based approach divides each users activities into training, validation, and test sets [12], [13]. The training set is used to define normal behavior while the validation set is compared to the training set and used to determine the

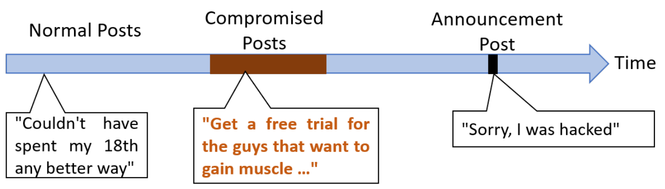

Normal Posts
Compromised Posts
Announcement Post
Time
"Couldn't have spent my 18th any better way"
"Get a free trial for the guys that want to gain muscle ..."
"Sorry, I was hacked"

Figure 1: A typical attack scenario of compromised account on Twitter. The original user will tweet before and after their account has been compromised, denoted as normal posts. When hackers take control of the account, they will publish tweets, i.e. compromised posts. When the user realizes the account was compromised, they will alert their followers of the compromise in an announcement post.

distance threshold indicating a different author. This approach tends to focus only on lexical features and ignores meta information, e.g. application used, which has been shown in compromised account analysis literature to indicate compromised accounts [4], [14]. The distance-based approach does not require prior knowledge of compromised accounts, because the validation sets includes posts from multiple users, i.e. which posts from different authors is known a priori.

User-level modeling summarizes each user's activities into a single vector and learns a lower-dimensional representation of all users. Anomalous users (i.e. compromised accounts) are detected either by training a classifier on this lower-dimensional representation or approximating the users' activities in the original feature space from the lower-dimensional representation. Karimi et al. learned the lower-dimensional representation by encoding users' posts with doc2vec and an LSTM, and trained a classifier on these encodings [15]. Reconstruction error is used to approximate users' activities from the lower-dimensional representation. Viswanath et al. learned a lower-dimensional representation of users' likes using PCA [7]. VanDam et al. applied autoencoders on multiple views of users's posts and learned a user representation from the views' shared latent space [8].

III. Problem Statement

Let $\mathcal{U} = \{u_1, u_2, \cdots, u_N\}$ be the set of social media users and $\mathcal{T} = \{T^{(1)}, T^{(2)}, \cdots, T^{(N)}$ be the set of all postings, where each $T^{(i)} = \{t_1^{(i)}, t_2^{(i)}, \cdots, t_{m_i}^{(i)}\}$ is the set of posts associated with user u_i's account. A user's account is compromised when an unauthorized person, i.e. hacker, gains access to the user's account and perform some actions, e.g. posting, from that account without the user's consent. We denote $y(t_j^{(i)})$ as the genuine author of the j-th post from user u_i's account.

Definition 3.1: **Compromised Post**: A post, $t^{(i)}$, from user u_i's account is said to be compromised if it was written by another user, i.e., $y(t^{(i)}) \neq u_i$.

Definition 3.2: **Compromised Account**: The social media account of user u_i, is said to be compromised if it is associated with at least one compromised post, i.e., $\exists t^{(i)} : y(t^{(i)}) \neq u_i$.

As proof of concept, this research focuses on compromised Twitter accounts, where each post corresponds to a user's tweet. The proposed framework can be easily generalized to other social media platforms or to other types of hackers' actions, e.g. liking or browsing. Figure 1 demonstrates a typical attack scenario of compromised accounts on Twitter. The genuine user initially publishes a series of tweets. When the account is compromised, the hacker will publish one or more tweets, shown in dark orange in Figure 1. When the original user discovers the account was compromised, he or she will perform actions to prevent the hacker from further posting, e.g., by changing password, and post an *announcement post*, shown in black, to inform their followers that the hackers' tweets were not written by the original user.

In this paper, we cast the compromised account detection problem as a binary classification task. Specifically, we classify each user-tweet pair, $(u_i, t^{(i)})$, either as positive class if $t^{(i)}$ is a compromised post or negative class if it is a genuine post. To predict the user-tweet pair, let $\mathbf{x}_t^{(i)} \in \mathbb{R}^d$ be the set of features associated with the post $t^{(i)}$ and $\mathbf{z}^{(i)} \in \mathbb{R}^p$ be the set of features associated with the user u_i. We further categorize the tweet features $\mathbf{x}_t^{(i)}$ into two types: *content* or *meta* tweet features. Content-related tweet features include the text message itself, hashtags, mentions, and URLs whereas tweet meta features include the location, source (application), and language used by the user to post the tweet. The tweet meta features are expected to have lower variability since a user will likely tweet from a limited number of applications (e.g., from a Web browser or via their smartphone app) and locations, usually in the same language. In contrast, content features such as the terms used, hashtags, and mentions may vary from one tweet to another. Such distinction between content and tweet meta features will be utilized by our proposed CAUTE framework. Finally, the user features $\mathbf{z}^{(i)}$ can be derived from the user profile or based on the user's initial posting behavior. The latter can be obtained by extracting features from the first k% tweets associated with the user. This is similar to the approach used in COMPA [4], [9].

IV. Proposed Framework

CAUTE is a 3-component neural network architecture that simultaneously learns the feature embedding of a given user-tweet pair (u, t) and uses the residual error of the embedding to determine the likelihood that user u is not the author of tweet t. A high likelihood would suggest that the tweet has a high probability of being a compromised post.

As noted in the Introduction, the raw user and tweet features, which are typically represented using one-hot encoding, are often too sparse to be effectively used for detecting compromised posts and accounts. To address this challenge, the first two components of our architecture, *tweet2user* and *user2tweet*, are feature encoders designed to learn nonlinear embeddings of the raw features, which are then used to approximate the user and tweet features respectively. The third component, *res2class*, uses the residual errors from the tweet2user and user2tweet encoders to predict whether the tweet was written by the user. The rationale behind our proposed framework is as follows. If user u is the author of tweet t, then the tweet features $\mathbf{x}_t$ can be used to predict the user features $\mathbf{z}$, and vice-versa. If the user is not the author of the tweet, then the residual errors of the predictions are likely to be large. A high-level schematic illustration of the CAUTE framework is shown in Figure 2.

A. *tweet2user Encoder*

The goal of the tweet2user encoder is to learn a nonlinear feature embedding of the tweet that can be used to predict the user features. One potential challenge to learning the embedding is that tweets with similar content can be posted by more than one user. To address this challenge, instead of learning the embedding from the tweet content features alone (e.g., the terms in the tweet), CAUTE also considers the tweet meta features to help identify the user who authored the tweet. The latent features derived by the tweet2user encoder thus represent not only the topics pertaining to the tweet content but also some information about the user who posted the tweet.

The tweet2user encoder is a feed-forward neural network, shown in the top left portion in Figure 2. Specifically, given a user-tweet pair, $(u_i, t^{(i)})$, with the corresponding user feature $\mathbf{z}^{(i)} \in \mathbb{R}^p$ and tweet feature $\mathbf{x}_t^{(i)} \in \mathbb{R}^d$, the network is trained to learn a pair of mapping functions $g_1 : \mathbb{R}^d \to \mathbb{R}^k$ and $g_2 : \mathbb{R}^k \to \mathbb{R}^p$ such that the following residual error:

$$\|g_2(g_1(\mathbf{x}_t^{(i)})) - \mathbf{z}^{(i)}\|^2 \qquad (1)$$

is minimized for all user-tweet pairs in the training data. The mapping function g_1 is implemented using a Leaky ReLU applied to the output of the first weight layer shown in Figure 2 while the mapping function g_2 is implemented using a linear activation function applied to the output of the second weight layer. The output of the Leaky ReLU unit, $g_1(\mathbf{x}_t^{(i)})$, thus provides a nonlinear embedding of the tweet features.

The input of the tweet2user encoder is a concatenation of the tweet content features $(\mathbf{x}_{t,c}^{(i)})$ and tweet meta features $(\mathbf{x}_{t,u}^{(i)})$, i.e., $\mathbf{x}_t^{(i)} = [\mathbf{x}_{t,c}^{(i)},\ \mathbf{x}_{t,u}^{(i)}]$. The output of tweet2user is trained to approximate the user features $\mathbf{z}^{(i)}$. How the user is represented, i.e. its raw features, can affect the user encoding accuracy of tweet2user. Ideally, the features should be unique for each user, i.e., no two users should have the same feature representation. A simple representation of users is a 1-hot encoded feature vector of length N, corresponding to the total number of users. However as social networks are continuously growing, this fixed-length feature vector would prevent CAUTE from learning the users who are not in the training set. Thus a fixed length user representation is derived from a profile of the user, e.g. a subset of their tweets.

B. *user2tweet Encoder*

As multiple users can post similar type of tweets, it is possible that the tweet2user encoder would predict the features of one user when the tweet is posted by a different user. To boost our confidence that a user is indeed the author of a tweet,

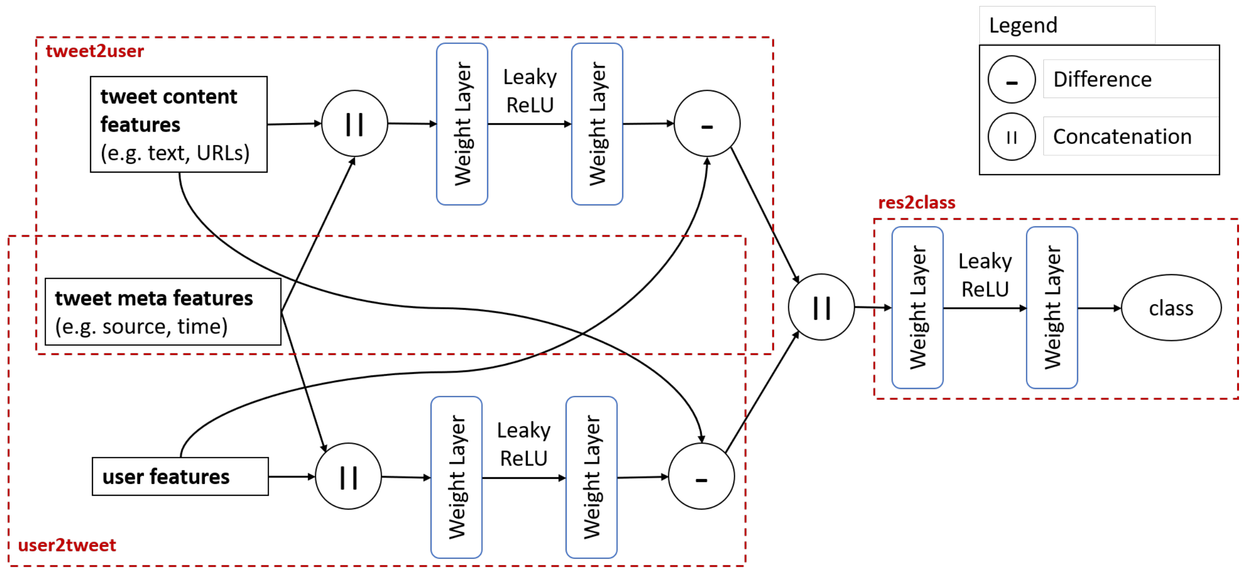


Figure 2: CAUTE Framework

a user2tweet encoder is simultaneously trained to recognize the type of tweets posted by the user. Since a user can post multiple tweets with different content, it is insufficient to use the user features alone as input to the user2tweet encoder as the encoder will always produce the same output given the same input features. In this case, the embedded features are likely to be the average feature representation of all the tweets by that same user. To overcome this limitation, the user2tweet encoder in CAUTE leverages the tweets' meta features to identify their content features. Specifically, the meta tweet features are concatenated with the user features before being provided as input of the user2tweet encoder, as shown in Figure 2. Formally, given a user-tweet pair, $(u_i, t^{(i)})$, with the corresponding user feature $\mathbf{z}^{(i)} \in \mathbb{R}^p$, tweet meta feature $\mathbf{x}_{t,u}^{(i)} \in \mathbb{R}^{d_1}$ and tweet content feature $\mathbf{x}_{t,c}^{(i)} \in \mathbb{R}^{d_2}$, the encoder is trained to learn a pair of mapping functions $h_1 : \mathbb{R}^{p+d_1} \to \mathbb{R}^k$ and $h_2 : \mathbb{R}^k \to \mathbb{R}^{d_2}$ such that the following residual error:

$$\|h_2(h_1(\mathbf{z}^{(i)}, \mathbf{x}_{t,u}^{(i)})) - \mathbf{x}_{t,c}^{(i)}\|^2 \quad (2)$$

is minimized for all user-tweet pairs. By optimizing the error function, the approximation will be close to the actual tweets' content features if the tweet is authored by the selected user. However if the tweets are authored by a different user, then their residual errors are likely to be large.

C. res2class Classifier

The tweet2user and user2tweet encoders provide approximations of the user and tweet features respectively. The third component of CAUTE, res2class, takes the residual errors of the two encoders as input to predict whether a tweet was authored by the corresponding user. The output of res2class is a positive class if the tweet was written by the user, and negative class if the tweet was written by a different user.

Given a user-tweet pair, $(u_i, t^{(i)})$, let $\Delta_{t2u}(u_i, t^{(i)}) = |g_2(g_1(\mathbf{x}_t^{(i)})) - \mathbf{z}^{(i)}| \in \mathbb{R}^p$ be the absolute residual error of tweet2user and $\Delta_{u2t}(u_i, t^{(i)}) = |h_2(h_1(\mathbf{z}^{(i)}, \mathbf{x}_{t,u}^{(i)})) - \mathbf{x}_{t,c}^{(i)}| \in \mathbb{R}^{d_2}$ be the corresponding absolute residual error of user2tweet. Absolute value is necessary to ensure res2class is focused on the magnitude of the residual rather than the direction of the error, i.e. overestimate or underestimate. The absolute residuals from the tweet2user and user2tweet encoders are concatenated and provided to a feed forward neural network with Leaky ReLu activation function. A cross entropy loss function was used to train the res2class classifier, enabling the output of res2class to represent the likelihood of the class.

V. Experimental Evaluation

This section describes the experiments conducted to evaluate the performance of CAUTE and other state-of-the-art algorithms for compromised account detection on a real-world Twitter dataset. We first describe data collected along with the features used to describe the users and their posts. We also discuss our approach for training the network before presenting the experimental results and discussion.

A. Data

We employed the Twitter streaming API to download the tweets posted between between April 27, 2015 and May 6, 2015. We extracted the IDs of users who posted the tweets and collected the 200 most recent tweets[2] by each user. As noted in [14], identifying compromised posts is a challenging annotation task. Following the approach used in Trang et al. [6], we artificially inserted compromised posts into the data by swapping posts from one user and assigning them to another user. With this approach, we created a dataset that

[2]Although we requested for 200 tweets, the number of actual tweets returned may vary since the user may delete some of their tweets before they were collected

contains tweets from 5524 users, where each user has posted, on average, about 173 'genuine' tweets. In addition, we artificially injected 957,392 'compromised' tweets by assigning each tweet to a random user. The entire dataset thus contains 1,917,342 (user, tweet) pairs with 50.1% pairs are genuine posts and the remaining 49.9% pairs are compromised.

Learning the encoding from a tweet's representation to its respective user's representation and vice versa is heavily reliant on the features used. Instead of manually identifying the discriminative features to be used for compromised account detection, CAUTE is designed to automatically learn the informative features using the tweet2user and user2tweet encoders. For each tweet, we consider the following raw tweet features:

- Content features:
  - Hashtags: hashtags that appear in the body of the tweet.
  - Mentions: which users were mentioned in the tweet.
  - URLs: domain name of any URLs in the tweet.
  - Text: terms that appear in the tweet.
- Meta features:
  - Time of Day: when was the tweet published.
  - Language: the language in which the tweet was written.
  - Source: the application used to post the tweet.

We used one-hot-encoding to represent each feature except for text. To reduce the sparsity of text features, we applied Singular Value Decomposition (SVD) with the number of components set to 30. To select this component set, we varied the number of components from 10 to 100 and measured the average pairwise distance between users and selected the number of components that produced a large pairwise distance.

As previously noted in the Introduction, in principle, CAUTE is applicable even in a cold-start scenario unlike other existing methods, e.g., by using the user profile information to create the user features. By applying CAUTE on the user and tweet features, we can detect whether the tweet is likely to be posted by the user. However, in practice, the detection performance under the cold-start scenario tends to be poor unless the user features are indicative of the type of tweets posted (e.g., if the user profile features include the topics of interest to the user). In our experiments, we reserve a subset of the initial posts made by each user to define the user features. Specifically, we applied SVD on the initial tweets and used their derived components as user features. The subset of tweets will be excluded from the dataset used to train the compromised detection model.

To ensure fairness in experimental evaluation, we reserved the same subset of the tweets made by each user to learn the behavioral profile of the users for all the baseline algorithms considered in this study. We explored two percentage values, 5% and 10% of the 200 tweets for each user, as our initial subset. Ideally, only a small percentage of tweets would be available to learn a user representation, even though having more tweets used to generate a profile would lead to better predictions of compromised posts.

B. Training a Neural Network

Neural network training is challenging due to the numerous hyperparameters that need to be tuned, including batch size and number of epochs. If the batch size is too large, the neural network over-generalizes the patterns learned, whereas a batch size that is too small leads to longer training time. We tested batch sizes of powers of 2 up to 2048 tweets and found that user-specific embeddings emerge when the batch size is either 16 or 32 tweets, without suffering a noticeable increase in training time. For these experiments, the tweet2user and user2tweet encoders were trained using batch size of 16 tweets. The res2class encoder focused on learning more general patterns, so batch size was set to 128 tweets.

In order for a neural network to be generalizable to unseen users, a sufficiently large sample of users is needed for training. However training on more users leads to a longer training time. To overcome this challenge, users were split equally into three sets for training, validation, and testing. Using all tweets from the users in the training set could still lead to long training time. However training on a random sample of their tweets could be just as effective as training on all of a users' tweets. After removing the tweets used to generate the user features, we selected 40% of the users' tweets to train the network, because it had similar performance to using 100% of the tweets. The model was evaluated on the withheld tweets from the training set users to determine when to stop training the neural network. If the loss increased by 0.1% for these tweets, training was terminated.

The datasets and their purposes are denoted as follows:

- Train: Approximately 1/3 of users and 40% of their tweets were used to train the neural networks. These tweets were also used to perform SVD in order to obtain the user features or their behavioral profile. The remaining 40% of tweets from these users determined how many epochs to train each neural network
- Validation: Approximately 1/3 of users that do not appear in the training set. This dataset was used for hyperparameter tuning, e.g. number of nodes in the neural network.
- Test: Remaining 1/3 of users. The (user, tweet) pairs for these users are used for evaluating the performance of the various algorithms.

For the validation and test set, the initial subset of tweets used to generate the SVD-based user feature vector were excluded. For training and validation sets, instances of compromised posts were created by matching each tweet with a random user that was not their respective user. That random user, assigned to be the hacker, was another user in the training or validation set respectively. For the test set, instances of compromised posts were created by matching all the tweets of a user in the test set to the same random user in the test set. Thus, for each user u_i in the test set, the tweets associated with u_i include all of their original tweets ('genuine' posts) as well as all the tweets from another user in the test set u_j ('compromised' posts) where $i \neq j$.

C. Experimental Results

In this subsection, we present the results of our experiments. In particular, we design the experiments to answer the following questions:

- How good is the feature embeddings learned by CAUTE?
- How well does CAUTE perform compared to existing methods for compromised post detection?
- What value does each component of CAUTE contribute to its overall performance?

1) Evaluating the Utility of CAUTE's Latent Features: The first research question is to evaluate the efficacy of using the features learned by CAUTE to detect compromised posts. Specifically, we use the feature embeddings learned from tweet2user and user2tweet encoders to train a classifier for predicting compromised posts. To do this, we first create a dataset containing a set of (user, tweet) pairs, where the user features correspond to the latent features of the user2tweet encoder while the tweet features are the latent features of the tweet2user encoder. Each (user, tweet) pair is assigned a class label of +1 if the tweet is a compromised post or -1 if the tweet is posted by the user. We evaluated the performance of CAUTE's latent features using both logistic regression and random forest as our classifiers.

We compare the performance of CAUTE's latent features against those generated by the following baselines:

- Raw features: In this approach, the user features are the same as the original user features provided as input to CAUTE (i.e., SVD applied to the text features of a subset of initial tweets posted by each user). For the tweet features, we applied one-hot encoding to the tweet content and meta features listed in Section VA. Due to memory limitations, we also applied SVD on the text features with number of components set to 30.
- Doc2vec: [16] This is a popular representation learning approach for documents. To create the user features, we first concatenated the text of a user's tweets into a single document. We repeated this process for each user to obtain 5524 documents. We then trained a doc2vec model on these documents to obtain the feature embedding for each user. To obtain the tweet features, we applied the doc2vec model to the text of each tweet.
- SVD: For user features, we use the same features as CAUTE. For tweet features, all the tweet content and meta features (e.g. source, text, language, etc.) were concatenated before applying SVD.

Table I summarizes the results of our experiment. We use Area under the ROC curve (AUC) for the positive class (compromised post) as our evaluation metric. For logistic regression, the results suggest that none of the baseline methods (raw features, doc2vec, and SVD) were able to produce classifiers that perform significantly better than random guessing (AUC score around 0.5). For random forest, doc2vec is the only baseline method that performs slightly better than random guessing, with an AUC score around 0.53.

Table I: Comparison of feature representation methods. Logistic regression and random forest classifiers are applied to detect compromised posts using the user and tweet features obtained from various methods. Results shown are evaluated in terms of AUC score for compromised posts.

Method	Logistic Regression	Random Forest
Raw Features	0.5036	0.5000
Doc2vec	0.5017	0.5353
SVD	0.5012	0.5022
Tweet2User & User2Tweet Features	**0.5964**	**0.5708**

These results showed the challenges of finding effective features for compromised post detection. Using the raw features alone is insufficient to produce classifiers that perform better than random guessing. The data contains too much variations that standard methods such as SVD and doc2vec were unable to create useful features that can accurately identify compromised posts. Transforming the features using a nonlinear encoder may lead to higher performance. Specifically training a random forest using doc2vec embeddings achieved slightly higher AUC than using the raw features or SVD. One rationale for this is that doc2vec tries to capture meaning within its embeddings. However, since the doc2vec features are learned in an unsupervised way, there are no guarantees that they would be effective for compromised post detection.

CAUTE's latent features derived from the tweet2user and user2tweet encoders were significantly better predictors for compromised post detection than the other baseline algorithms. The results hold for both logistic regression and random forest classifiers. Given that the tweet2user and user2tweet encoders are designed to recognize tweets coming from the genuine users, their embeddings of tweets and users should be good predictors of whether a post is compromised.

2) Performance Comparison: Next, we evaluate CAUTE's overall performance as a whole and show that CAUTE can detect compromised posts better than existing methods. We consider both the overall performance, weighing false positive and false negatives equally, as well as early detection of compromised accounts. We compare CAUTE against the following state-of-the-art algorithms;

- COMPA: We applied the Trang et al. [6] adaptation of the COMPA algorithm [9], [4], which builds a profile for each user based on their tweets' features. Previously unseen tweets are compared to the profile to determine their likelihood of being anomalous, i.e. assigned an anomaly score. The tweet's anomaly score is the weighted average of the anomaly score of each of its features.
- Principal Component Analysis (PCA): This is an unsupervised anomaly detection approach developed in [7] to extract principal components of the data and then reconstruct the original features from the principal components.

Table II: Comparison of compromised post detection between CAUTE and the baseline algorithms; COMPA and PCA. Algorithms are evaluated in terms of their AUC score. The number of posts used to generate the user features varied from 5% of their posts to 10%. CAUTE consistently outperforms the baselines by at least 2%. All of the algorithms improve performance when more tweets are used to generate the user features.

Algorithm	5% Tweets	10% Tweets
COMPA	0.6415	0.6779
PCA	0.5064	0.5088
CAUTE	**0.6707**	**0.7014**

The likelihood of a compromised post is measured by its reconstruction error. In this experiment, we concatenate the users' terms (from the tweets used to create user features) with the tweet features of their tweets. Next we learn the principal components of this matrix for users in the training set. From the learned principal components, we transform user-tweet pairs and calculate their reconstruction error.

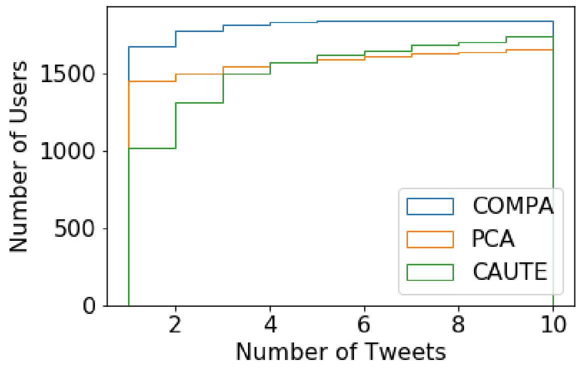
CAUTE is consistently better at detecting compromised posts than the baseline algorithms, as shown in Table II. PCA has the lowest AUC, comparable to random guessing, which is not surprising given the difficulty of the detection task and the unsupervised nature of the algorithm. COMPA and CAUTE were significantly better than PCA. Additionally, both CAUTE and COMPA were better at detecting whether a post belonged to the genuine user when more posts were used to generate the user representation.

In addition to detecting whether a tweet was published by its respective author, it is crucial to detecting the compromised accounts early. In this experiment, we provide each user with a set of compromised posts and test how many such posts are observed before one of them is flagged as compromised. Ideally, the first post should be detected as compromised, to mitigate any further damages posed by the hacker.

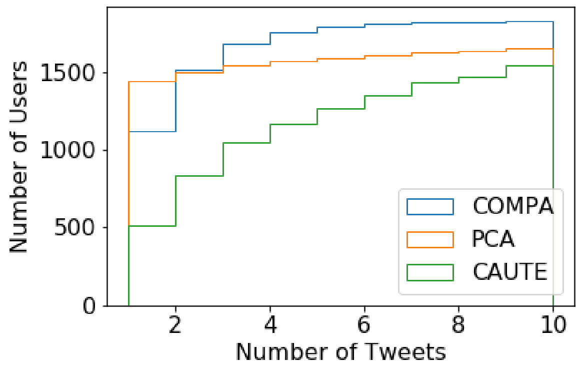
On average, COMPA detects compromised posts earlier than CAUTE. However, given its lower AUC, it also classifies more posts as false positives, i.e. genuine posts classified as compromised post. We observe that COMPA detects the first message as compromised for the majority of the 1841 users in the test set, shown in Figure 3a. CAUTE is more conservative when flagging a post as compromised. Within the first 10 tweets, CAUTE can detect most compromised accounts.

To verify that these algorithms were not predicting genuine posts as compromised, we also provided each user with their own posts, and tested how many posts before each algorithm would identify one of them as compromised. Both PCA and COMPA mis-classify genuine users as compromised early, shown in Figure 3b. PCA mislabels 1447 of the 1841 users as compromised from their first tweet. COMPA incorrectly predicts only 1125 users as compromised from their first tweet. However, by their 10th tweet, COMPA incorrectly predicts 1832 users as compromised. CAUTE has significantly fewer mis-classifications on the first tweet than the other algorithms. It only incorrectly classifies 520 genuine users as compromised based on their first tweet. By the users' 10th tweet, CAUTE incorrectly identified only 1572 genuine users as compromised.

Thus, while COMPA and PCA can detect a compromised account earlier than CAUTE, it is at the cost of incorrectly classifying most users' genuine posts as compromised. CAUTE detects compromised accounts slightly later, e.g. from the 3rd compromised tweet. However, CAUTE also detects the genuine users' posts correctly more often than COMPA or PCA, thus raising fewer false alarms. With fewer false alarms, the user is more likely to take actions to mitigate the attack, e.g. changing their password, when a compromise is detected.



(a) Compromised accounts detected



(b) Genuine accounts predicted as compromised

Figure 3: Measurement of how many posts from each user are observed before one of them is flagged as compromised. In (a), all of the tweets provided to the user were compromised. In (b), all of the tweets were their own tweets. COMPA detects most compromised accounts from the first tweet, but also predicts genuine tweets as compromised at a higher rate. CAUTE detects most compromised accounts within the first 10 tweets, and has a significantly lower false positive rate from the the genuine users' posts.

Table III: AUC of the tweet2user and user2tweet encoders in comparison to CAUTE. For tweet2user and user2tweet embedders, tweet-user pairs were scored based on the sum of absolute residuals. Tweet2user encoder was better than the user2tweet encoder at identifying whether a tweet matched its respective user. Both encoders provided some information to CAUTE, which achieved higher AUC than either of the individual encoders.

Component	AUC
tweet2user	0.6212
user2tweet	0.5581
tweet2user + res2class	0.6419
user2tweet + res2class	0.5772
res2class (freeze residuals)	0.6691
CAUTE	0.6707

3) Evaluating the Importance of Each Component: Next we analyze the importance of each component of CAUTE. This analysis considers how the removal of one or two of the components of CAUTE affects performance, shown in Table III. The tweet2user encoder is better at detecting tweets from a hacker than the user2tweet encoder. Despite appending some tweet information to the user in the user2tweet encoder, it still had lower performance. The inclusion of the res2class encoder yields significantly higher AUC for both encoders. This is expected because the res2class is trained on the specific classification task, identifying whether a tweet was written by their respective user, whereas the other components are focused on minimizing the approximation error. The addition of the user2tweet embedder into CAUTE yields slightly higher performance than using the tweet2user and res2class encoders.

To train the res2class component of CAUTE, the residuals could either be frozen or updated during back propagation. If the residuals are frozen, res2class only learns the neuron weights between the residual and the class. In CAUTE, the residuals are updated as back propagation would update the weights of the user2tweet and tweet2user encoders as well as the res2class encoder. The pretrained tweet2user and user2tweet encoders are passed to CAUTE. Frozen residuals is significantly faster at converging, as there are fewer weights that need updating. However allowing CAUTE to apply back propagation through the entire network yields slightly higher performance, as evident in Table III.

VI. Conclusion

This paper proposed CAUTE, a compromised account user-tweet encoder which can identify tweets that do not belong to the designated user, i.e. are compromised. CAUTE is able to identify tweets matched with the incorrect user, from a subset of each user's tweets. Additionally, CAUTE can identify whether tweets belong to users it had not previously observed. In future work, we plan to extend our framework to different types of compromised accounts. Whereas CAUTE focused encoding users and tweets, future work would encode either users with other users to identify suspicious following activity or users with their social media activity patterns to identify hackers seeking information about the compromised user.

Acknowledgement

This work is supported in part by the U.S. National Science Foundation under grant #IIS-1615612.

References

[1] K. Olmstead and A. Smith, "Americans and cybersecurity," Pew Research CCenter, Tech. Rep., 2017.

[2] K. Thomas, F. Li, C. Grier, and V. Paxson, "Consequences of connectivity: Characterizing account hijacking on twitter," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '14, 2014.

[3] S. Ovide, "False ap twitter message sparks stock-market selloff," *Wall Street Journal*, 4 2013. [Online]. Available: https://www.wsj.com/articles/SB10001424127887323735604578440971574897016

[4] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna, "COMPA: Detecting Compromised Accounts on Social Networks," in *ISOC Network and Distributed System Security Symposium (NDSS)*. San Diego, California: Internet Society, 2013.

[5] M. Nauta, "Detecting Hacked Twitter Accounts by Examining Behavioural Change using Twitter Metadata," in *Proceedings of the 25th Twente Student Conference on IT*, 2016.

[6] D. Trang, F. Johansson, and M. Rosell, "Evaluating algorithms for detection of compromised social media user accounts," in *2015 Second European Network Intelligence Conference*, Sept 2015, pp. 75–82.

[7] B. Viswanath, M. A. Bashir, M. Crovella, S. Guha, K. P. Gummadi, B. Krishnamurthy, and A. Mislove, "Towards detecting anomalous user behavior in online social networks," in *23rd USENIX Security Symposium (USENIX Security 14)*. USENIX Association, 2014.

[8] C. VanDam, P.-N. Tan, J. Tang, and H. Karimi, "Cadet: Compromised account detection using unsupervised learning," in *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ser. ASONAM 2018, 2018.

[9] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna, "Towards detecting compromised accounts on social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 4, pp. 447–460, 2017.

[10] J. Bohacik, A. Fuchs, and M. Benedikovic, "Detecting compromised accounts on the pokec online social network," in *2017 International Conference on Information and Digital Technologies (IDT)*. Zilina, Slovakia: IEEE, 2017, pp. 56–60.

[11] X. Ruan, Z. Wu, H. Wang, and S. Jajodia, "Profiling Online Social Behaviors for Compromised Account Detection," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 1, pp. 176–187, 2016.

[12] R. A. Igawa, A. M. G. de Almeida, B. B. Zarpelao, and S. Barbon, Jr, "Recognition of compromised accounts on twitter," in *Proceedings of the Annual Conference on Brazilian Symposium on Information Systems: Information Systems: A Computer Socio-Technical Perspective - Volume 1*, ser. SBSI 2015. New York, NY, USA: ACM, 2015.

[13] S. Barbon, Jr, R. A. Igawa, and B. Bogaz Zarpelão, "Authorship verification applied to detection of compromised accounts on online social networks," *Multimedia Tools Appl.*, vol. 76, no. 3, Feb. 2017.

[14] C. VanDam, J. Tang, and P.-N. Tan, "Understanding compromised accounts on twitter," in *Proceedings of the International Conference on Web Intelligence*, ser. WI '17, 2017.

[15] H. Karimi, C. VanDam, L. Ye, and J. Tang, "End-to-end compromised account detection," in *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ser. ASONAM 2018, Barcelona, Spain, 2018.

[16] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ser. ICML'14, 2014.